Classifier Combination for Contextual Idiom Detection Without Labelled Data

Linlin Li and **Caroline Sporleder**
Saarland University
Postfach 15 11 50
66041 Saarbrücken
Germany
{linlin,csporled}@coli.uni-saarland.de

Abstract

We propose a novel unsupervised approach for distinguishing literal and non-literal use of idiomatic expressions. Our model combines an unsupervised and a supervised classifier. The former bases its decision on the cohesive structure of the context and labels training data for the latter, which can then take a larger feature space into account. We show that a combination of both classifiers leads to significant improvements over using the unsupervised classifier alone.

1 Introduction

Idiomatic expressions are abundant in natural language. They also often behave idiosyncratically and are therefore a significant challenge for natural language processing systems. For example, idioms can violate selectional restrictions (as in *push one's luck*), disobey typical subcategorisation constraints (e.g., *in line* without a determiner before *line*), or change the default assignments of semantic roles to syntactic categories (e.g., in *break sth with X* the argument *X* would typically be an instrument but for the idiom *break the ice* it is more likely to fill a patient role, as in *break the ice with Russia*).

In order to deal with such idiosyncracies and assign the correct analyses, NLP systems need to be able to recognise idiomatic expressions. Much previous research on idioms has been concerned with *type-based classification*, i.e., dividing expressions into 'idiom' or 'not idiom' irrespective of their actual use in a given context. However, while some expressions, such as *by and large*, always have an idiomatic meaning, several other expressions, such as *break the ice* or *spill the beans*, can be used literally as well as idiomatically (see examples (1) and (2), respectively). Sometimes the literal usage can even dominate in a domain, as for *drop the ball*, which occurs fairly frequently in a literal sense in the sports section of news texts.

(1) Dad had to break the ice on the chicken troughs so that they could get water.

(2) Somehow I always end up spilling the beans all over the floor and looking foolish when the clerk comes to sweep them up.

Hence, whether a particular occurrence of a potentially ambiguous expression has literal or non-literal meaning has to be inferred from the context (*token-based idiom classification*). Recently, there has been increasing interest in this classification task and both supervised and unsupervised techniques have been proposed. The work we present here builds on previous research by Sporleder and Li (2009), who describe an unsupervised method that exploits the presence or absence of cohesive ties between the component words of a potential idiom and its context to distinguish between literal and non-literal use. If strong ties can be found the expression is classified as literal otherwise as non-literal. While this approach often works fairly well, it has the disadvantage that it focuses exclusively on lexical cohesion, other linguistic cues that might influence the classification decision are disregarded.

We show that it is possible to improve on Sporleder and Li's (2009) results by employing a two-level strategy, in which a cohesion-based unsupervised classifier is combined with a supervised classifier. We use the unsupervised classifier to label a sub-set of the test data with high confidence. This sub-set is then passed on as training data to the supervised classifier, which then labels the remainder of the data set. Compared to a fully unsupervised approach, this two-stage method has the advantage that a larger feature set can be exploited. This is beneficial for examples, in which the cohesive ties are relatively weak but which contain other linguistic cues for literal or non-literal use.

Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pages 315–323,
Singapore, 6-7 August 2009.

2 Related Work

Most studies on idiom classification focus on type-based classification; few researchers have worked on token-based approaches (i.e., classification of an expression in a given context). Type-based methods frequently exploit the fact that idioms have a number of properties which differentiate them from other expressions. For example, they often exhibit a degree of syntactic and lexical fixedness. Some idioms, for instance, do not allow internal modifiers (**shoot the long breeze*) or passivisation (**the bucket was kicked*). They also typically only allow very limited lexical variation (**kick the vessel*, **strike the bucket*).

Many approaches for identifying idioms focus on one of these two aspects. For instance, measures that compute the association strength between the elements of an expression have been employed to determine its degree of compositionality (Lin, 1999; Fazly and Stevenson, 2006) (see also Villavicencio et al. (2007) for an overview and a comparison of different measures). Other approaches use Latent Semantic Analysis (LSA) to determine the similarity between a potential idiom and its components (Baldwin et al., 2003). Low similarity is supposed to indicate low compositionality. Bannard (2007) looks at the syntactic fixedness of idiomatic expressions, i.e., how likely they are to take modifiers or be passivised, and compares this to what would be expected based on the observed behaviour of the component words. Fazly and Stevenson (2006) combine information about syntactic and lexical fixedness (i.e., estimated degree of compositionality) into one measure.

The few token-based approaches include a study by Katz and Giesbrecht (2006), who devise a supervised method in which they compute the meaning vectors for the literal and non-literal usages of a given expression in the training data. An unseen test instance of the same expression is then labelled by performing a nearest neighbour classification.

Birke and Sarkar (2006) model literal vs. non-literal classification as a word sense disambiguation task and use a clustering algorithm which compares test instances to two automatically constructed seed sets (one with literal and one with non-literal expressions), assigning the label of the closest set. While the seed sets are created without immediate human intervention they do rely on manually created resources such as databases of known idioms.

Cook et al. (2007) and Fazly et al. (2009) propose an alternative method which crucially relies on the concept of *canonical form*, which is a fixed form (or a small set of those) corresponding to the syntactic pattern(s) in which the idiom normally occurs (Riehemann, 2001).[1] The canonical form allows for inflectional variation of the head verb but not for other variations (such as nominal inflection, choice of determiner etc.). It has been observed that if an expression is used idiomatically, it typically occurs in its canonical form. For example, Riehemann (2001, p. 34) found that for decomposable idioms 75% of the occurrences are in canonical form, rising to 97% for non-decomposable idioms.[2] Cook et al. exploit this behaviour and propose an unsupervised method which classifies an expression as idiomatic if it occurs in canonical form and literal otherwise.

Finally, in earlier work, we proposed an unsupervised method which detects the presence or absence of cohesive links between the component words of the idiom and the surrounding discourse (Sporleder and Li, 2009). If such links can be found the expression is classified as 'literal' otherwise as 'non-literal'. In this paper we show that the performance of such a classifier can be significantly improved by complementing it with a second-stage supervised classifier.

3 First Stage: Unsupervised Classifier

As our first-stage classifier, we use the unsupervised model proposed by Sporleder and Li (2009). This model exploits the fact that words in a coherent discourse exhibit *lexical cohesion* (Halliday and Hasan, 1976), i.e. concepts referred to in sentences are typically related to other concepts mentioned elsewhere in the discourse. Given a suitable measure of semantic relatedness, it is possible to compute the strength of such cohesive ties between pairs of words. While the component words of literally used expressions tend to exhibit lexical cohesion with their context, the words of non-literally used expressions do not. For example, in (3) the expression *play with fire* is used literally and the word *fire* is related to surrounding words like *grilling*, *dry-heat*, *cooking*, and *coals*. In (4), however *play with fire* is used non-literally and cohesive ties be-

[1] This is also the form in which an idiom is usually listed in a dictionary.

[2] Decomposable idioms are expressions such as *spill the beans* which have a composite meaning whose parts can be mapped to the words of the expression (e.g., *spill*→'reveal', *beans*→'secret').

tween *play* or *fire* and the context are absent.

(3) **Grilling** outdoors is much more than just another **dry-heat cooking** method. It's the chance to play with **fire**, satisfying a primal urge to stir around in **coals** .

(4) And PLO chairman Yasser Arafat has accused Israel of playing with fire by supporting HAMAS in its infancy.

To determine the strength of cohesive links, the unsupervised model builds a graph structure (called *cohesion graph*) in which all pairs of content words in the context are connected by an edge which is weighted by the pair's semantic relatedness. Then the *connectivity* of the graph is computed, defined as the average edge weight. If the connectivity increases when the component words of the idiom are removed, then there are no strong cohesive ties between the expression and the context and the example is labelled as 'non-literal', otherwise it is labelled as 'literal'.

To model semantic distance, we use the *Normalized Google Distance* (NGD, see Cilibrasi and Vitanyi (2007)), which computes relatedness on the basis of page counts returned by a search engine.[3] It is defined as follows:

$$NGD(x,y) = \frac{max\{log\ f(x), log\ f(y)\} - log\ f(x,y)}{log\ M - min\{log\ f(x), log\ f(y)\}} \quad (5)$$

where x and y are the two words whose association strength is computed (e.g., *fire* and *coal*), $f(x)$ is the page count returned by the search engine for x (and likewise for $f(y)$ and y), $f(x,y)$ is the page count returned when querying for "x AND y", and M is the number of web pages indexed by the search engine. The basic idea is that the more often two terms occur together, relative to their overall occurrence, the more closely they are related.

We hypothesise that the unsupervised classifier will give us relatively good results for some examples. For instance, in (3) there are several strong cues which suggest that *play with fire* is used literally. However, because the unsupervised classifier only looks at lexical cohesion, it misses many other clues which could help distinguish literal and non-literal usages. For example, if *break the ice* is followed by the prepositions *between* or *over* as in example (6), it is more likely to be used idiomatically (at least in the news domain).

(6) "Gujral will meet Sharif on Monday and discuss bilateral relations," the Press Trust of India added. The minister said Sharif and Gujral would be able to break the ice over Kashmir.

Furthermore, idiomatic usages also exhibit cohesion with their context but the cohesive ties are with the *non-literal* meaning of the expression. For example, in news texts, *break the ice* in its figurative meaning often co-occurs with *discuss*, *relations*, *talks* or *diplomacy* (see (6)). At the moment we do not have any way to model these cohesive links, as we do not know the non-literal meaning of the idiom.[4] However if we had labelled data we could train a supervised classifier to learn these and other contextual clues. The trained classifier might then be able to correctly classify examples which were misclassified by the unsupervised classifier, i.e., examples in which the cohesive ties are weak but where other clues exist which indicate how the expression is used.

For example, in (7) there is weak cohesive evidence for a literal use of *break the ice*, due to the semantic relatedness between *ice* and *water*. However, there are stronger cues for non-literal usage, such as the preposition *between* and the presence of words like *diplomats* and *talks*, which are indicative of idiomatic usage. Examples like this are likely to be misclassified by the unsupervised model; a supervised classifier, on the other hand, has a better chance to pick up on such additional cues and predict the correct label.

(7) Next week the two diplomats will meet in an attempt to break the ice between the two nations. A crucial issue in the talks will be the long-running water dispute.

4 Second Stage: Supervised Classifier

For the supervised classifier, we used Support Vector Machines as implemented by the LIBSVM package.[5] We implemented four types of features, which encode both cohesive information and word co-occurrence more generally.[6]

[3] We employ Yahoo! rather than Google since we found that it returns more stable counts.

[4] It might be possible to compute the Normalized Google Distance between the whole expression and the words in the context, assuming that whenever the whole expression occurs it is much more likely to be used figuratively than literally. For expressions in canonical form this is indeed often the case (Riehemann, 2001), however there are exceptions (see Section 6.1) for which such an approach would not work.

[5] Available from: `http://www.csie.ntu.edu.tw/~cjlin/libsvm/` We used the default parameters.

[6] We also experimented with linguistically more informed features, such as the presence of named entities in the local context of the expression, and properties of the subject or co-ordinated verbs, but we found that these features did not lead to a better performance of the supervised classifier. This is probably partly due to data sparseness.

Salient Words (salW) This feature aims to identify words which are particularly *salient* for literal usage. We used a frequency-based definition of salience and computed the *literal saliency score* for each word in a five-paragraph context around the target expression:

$$sal_{lit}(w) = \frac{\log f_{lit}(w) \times i_{lit}(w)}{\log f_{nonlit}(w) \times i_{nonlit}(w)} \quad (8)$$

where $sal_{lit}(w)$ is the saliency score of the word w for the class lit; $f_{lit}(w)$ is the token frequency of the word w for literally used expressions; $i_{lit}(w)$ is the number of instances of the target expressions classified as lit which co-occur with word w (and mutatis mutandis $nonlit$ for target expressions labelled as non-literal).[7]

Words with a high sal_{lit} occur much more frequently with literal usages than with non-literal ones. Conversely, words with a low sal_{lit} should be more indicative of the non-literal class. However, we found that, in practice, the measure is better at picking out indicative words for the literal class; non-literal usages tend to co-occur with a wide range of words. For example, among the highest scoring words for *break the ice* we find *thick, bucket, cold, water, reservoir* etc. While we do find words like *relations, diplomacy, discussions* among the lowest scoring terms (i.e., terms indicative of the non-literal class), we also find a lot of noise (*ask, month*). The effect is even more pronounced for other expressions (like *drop the ball*) which tend to be used idiomatically in a wider variety of situations (*drop the ball on a ban of chemical weapons*, *drop the ball on debt reduction* etc.).

We implement the saliency score in our model by encoding for the 300 highest scoring words whether the word is present in the context of a given example and how frequently it occurs.[8] Note that this feature (as well as the next one) can be computed in a per-idiom or a generic fashion. In the former case, we would encode the top 300 words separately for each idiom in the training set, in the latter across all idioms (with the consequence that more frequent idioms in the training set contribute to more positions in the feature vector). We found that, in practice, it does not make a big difference which variant is used. Moreover, in our bootstrapping scenario, we cannot ensure that we have sufficient examples of each idiom in the training set to train separate classifiers, so we opted for generic models throughout all experiments.

Related Words (relW) This feature set is a variant of the previous one. Here we score the words not based on their saliency but we determine the semantic relatedness between the noun in the idiomatic expression and each word in the global context, using the *Normalized Google Distance* mentioned in Section 3. Again we encode the 300 top-scoring words.

While the *related words* feature is less prone to overestimation of accidental co-occurrence than the saliency feature, it has the disadvantage of conflating different word senses. For example, among the highest scoring words for *ice* are *cold, melt, snow, skate, hockey* but also *cream, vanilla, dessert*.

Relatedness Score (relS) The fourth feature set implements the *relatedness score* which encodes the scores for the 100 most highly weighted edges in the cohesion graph of an instance.[9] If these scores are high, there are many cohesive ties with the surrounding discourse and the target expression is likely to be used literally.

Discourse Connectivity (connect.) Finally, we implemented two features which look at the cohesion graph of an instance. We encode the connectivity of the graph (i) when the target expression is included and (ii) when it is excluded. The unsupervised classifier uses the difference between these two values to make its prediction. By encoding the absolute connectivity values as features we enable the supervised classifier to make use of this information as well.

5 Combining the Classifiers

As mentioned before, we use the unsupervised classifier to label an initial training set for the supervised one. To ensure that the training set does not contain too much noise, we only add those examples about which the unsupervised classifier is

[7] Our definition of sal_{lit} bears similarities with the well known $tf.idf$ score. We include both the term frequencies (f_{lit}) and the instance frequencies (i_{lit}) in the formula because we believe both are important. However, the instance frequency is more informative and less sensitive to noise because it indicates that expression classified as 'literal' *consistently* co-occurs with the word in question. Therefore we weight down the effect of the term frequency by taking its log.

[8] We also experimented with different feature dimensions besides 300 but did not find a big difference in performance.

[9] We only used the 100 highest ranked edges because we are looking at a specific context here rather than the contexts of the literal or non-literal class overall. Since the contexts we use are only five paragraphs long, recording the 100 strongest edges seems sufficient.

most confident. We thus need to address two questions: (i) how to define a *confidence function* for the unsupervised classifier, and (ii) how to set the *confidence threshold* governing what proportion of the data set is used for training the second classifier.

The first question is relatively easy to answer: as the unsupervised classifier bases its decision on the difference in connectivity between including or excluding the component words of the idiom in the cohesion graph, an obvious choice for a confidence function is the difference in connectivity; i.e., the higher the difference, the higher the confidence of the classifier in the predicted label.

The confidence threshold could be selected on the basis of the unsupervised classifier's performance on a development set. Note that when choosing such a threshold there is usually a trade-off between the size of the training set and the amount of noise in it: the lower the threshold, the larger and the noisier the training set. Ideally we would like a reasonably-sized training set which is also relatively noise-free, i.e., does not contain too many wrongly labelled examples. One way to achieve this is to start with a relatively small training set and then expand it gradually.

A potential problem for the supervised classifier is that our data set is relatively imbalanced, with the non-literal class being four times as frequent as the literal class. Supervised classifiers often have problems with imbalanced data and tend to be overly biased towards the majority class (see, e.g., Japkowicz and Stephen (2002)). To overcome this problem, we experimented with boosting the literal class with additional examples.[10] We describe our methods for training set enlargement and boosting the literal class in the remainder of this section.

Iteratively Enlarging the Training Set A typical method for increasing the training set is to go through several iterations of enlargement and retraining.[11] We adopt a conservative enlargement strategy: we only consider instances on whose labels both classifiers agree and we use the confidence function of the unsupervised classifier to determine which of these examples to add to the training set. The motivation for this is that we hypothesise that the supervised classifier will not have a very good performance initially, as it is trained on a very small data set. As a consequence its confidence function may also not be very accurate. On the other hand, we know from Sporleder and Li (2009) that the unsupervised classifier has a reasonably good performance. So while we give the supervised classifier a veto-right, we do not allow it to select new training data by itself or overturn classifications made by the unsupervised classifier.

A similar strategy was employed by Ng and Cardie (2003) in a self-training set-up. However, while they use an ensemble of supervised classifiers, which they re-train after each iteration, we can only re-train the second classifier; the first one, being unsupervised, will never change its prediction. Hence it does not make sense to go through a large number of iterations; the more iterations we go through, the closer the performance of the combined classifier will be to that of the unsupervised one because that classifier will label a larger and larger proportion of the data. However, going through one or two iterations allows us to slowly enlarge the training set and thereby gradually improve the performance of the supervised classifier.

In each iteration, we select 10% of the remaining examples to be added to the training set.[12] We could simply add those 10% of the data about which the unsupervised classifier is most confident, but if the classifier was more confident about one class than about the other, we would risk obtaining a severely imbalanced training set. Hence, we decided to separate examples classified as 'literal' from those classified as 'non-literal' and add the top 10% from each set. Provided the automatic classification is reasonably accurate, this will ensure that the distribution of classes in the training set is roughly similar to that in the overall data set at least at the early stages of the bootstrapping.

Boosting the Literal Class As the process goes on, we are still likely to introduce more and more imbalance in the training set. This is due to the fact that the supervised classifier is likely to have some bias towards the majority class (and our experiments in Section 6.2 suggest that this is indeed the case). Hence, as the bootstrapping process goes on, potentially more and more examples will be labelled as 'non-literal' and if we always select the top 10% of these, our training set will gradually

[10] Throughout this paper, we use the term 'boosting' in a non-technical sense.

[11] In our case re-training also involves re-computing the ranked lists of salient and related words. As the process goes on the classifier will be able to discover more and more useful cue words and encode them in the feature vector.

[12] Since we do not have a separate development set, we chose the value of 10% intuitively as it seemed a reasonably good threshold.

become more imbalanced. This is a well-known problem for bootstrapping approaches (Blum and Mitchell, 1998; Le et al., 2006). We could counteract this by selecting a higher proportion of examples labelled as 'literal'. However given that the number of literal examples in our data set is relatively small, we would soon deplete our literal instance pool and moreover, because we would be forced to add less confidently labelled examples for the literal class, we are likely to introduce more noise in the training set.

A better option is to boost the literal class with external examples. To do this we exploit the fact that non-canonical forms of idioms are highly likely to be used literally. Given that our data set only contains canonical forms (see Section 6.1), we automatically extract non-canonical form variants and label them as 'literal'. To generate possible variants, we either (i) change the number of the noun (e.g., *rock the boat* becomes *rock the boats*), (ii) change the determiner (e.g., *rock a boat*), or (iii) replace the verb or noun by one of its synonyms, hypernyms, or siblings from WordNet (e.g., *rock the ship*). While this strategy does not give us additional literal examples for all idioms, for example we were not able to find non-canonical form occurrences of *sweep under the carpet* in the Gigaword corpus, for most idioms we were able to generate additional examples. Note that this data set is potentially noisy as not all non-canonical form examples are used literally. However, when checking a small sample manually, we found that only very small percentage ($<< 1\%$) was mis-labelled.

To reduce the classifier bias when enlarging the training set, we add additional literal examples during each iteration to ensure that the class distribution does not deviate too much from the distribution originally predicted by the unsupervised classifier.[13] The examples to be added are selected randomly but we try to ensure that each idiom is represented. When reporting the results, we disregard these additional external examples.

6 Experiments and Results

We carried out a number of different experiments. In Section 6.2 we investigate the performance of the different features of the supervised classifier and in Section 6.3 we look more closely at the behaviour of the combined classifier. We start by describing the data set.

6.1 Data

We used the data from Sporleder and Li (2009), which consist of 17 idioms that can be used both literally and non-literally (see Table 1). For each expression, all canonical form occurrences were extracted from the Gigaword corpus together with five paragraphs of context and labelled as 'literal' or 'non-literal'.[14] The inter-annotator agreement on a small sample of doubly annotated examples was 97% and the kappa score 0.7 (Cohen, 1960).

expression	literal	non-literal	all
back the wrong horse	0	25	25
bite off more than one can chew	2	142	144
bite one's tongue	16	150	166
blow one's own trumpet	0	9	9
bounce off the wall*	39	7	46
break the ice	20	521	541
drop the ball*	688	215	903
get one's feet wet	17	140	157
pass the buck	7	255	262
play with fire	34	532	566
pull the trigger*	11	4	15
rock the boat	8	470	478
set in stone	9	272	281
spill the beans	3	172	175
sweep under the carpet	0	9	9
swim against the tide	1	125	126
tear one's hair out	7	54	61
all	862	3102	3964

Table 1: Idiom statistics (* indicates expressions for which the literal usage is more common than the non-literal one)

6.2 Feature Analysis for the Supervised Classifier

In a first experiment, we tested the contribution of the different features (Table 2). For each set, we trained a separate classifier and tested it in 10-fold cross-validation mode. We also tested the performance of the first three features combined (salient and related words and relatedness score) as we wanted to know whether their combination leads to performance gains over the individual classifiers. Moreover, testing these three features in combination allows us to assess the contribution of the connectivity feature, which is most closely related to the unsupervised classifier. We report the accuracy, and because our data are fairly imbalanced,

[13] We are assuming that the true distribution is not known and use the predictions of the unsupervised classifier to approximate the true distribution.

[14] The restriction to canonical forms was motivated by the fact that for the mostly non-decomposable idioms in the set, the vast majority (97%) of non-canonical form occurrences will be used literally (see Section 2).

also the F-Score for the minority class ('literal').

Feature	Avg. literal (%) Prec.	Rec.	F-Score	Avg. (%) Acc.
salW	77.10	56.10	65.00	86.83
relW	78.00	43.20	55.60	84.99
relS	74.90	37.50	50.00	83.68
connectivity	78.30	2.10	4.10	78.58
salW+relW+relS	82.90	63.50	71.90	89.20
all	85.80	66.60	75.00	90.34

Table 2: Performance of different feature sets, 10-fold cross-validation

It can be seen that the *salient words* (*salW*) feature has the highest performance of the individual features, both in terms of accuracy and in terms of literal F-Score, followed by *related words* (*relW*), and *relatedness score* (*relS*). Intuitively, it is plausible that the saliency feature performs quite well as it can also pick up on linguistic indicators of idiom usage that do not have anything to do with lexical cohesion. However, a combination of the first three features leads to an even better performance, suggesting that the features do indeed model somewhat different aspects of the data.

The performance of the connectivity feature is also interesting: while it does not perform very well on its own, as it over-predicts the non-literal class, it noticeably increases the performance of the model when combined with the other features, suggesting that it picks up on complementary information.

6.3 Testing the Combined Classifier

We experimented with different variants of the combined classifier. The results are shown in Table 3. In particular, we looked at: (i) combining the two classifiers without training set enlargement or boosting of the literal class (*combined*), (ii) boosting the literal class with 200 automatically labelled non-canonical form examples (*combined+boost*), (iii) enlarging the training set by iteration (*combined+it*), and (iv) enlarging the training set by iteration and boosting the literal class after each iteration (*combined+boost+it*). The table shows the literal precision, recall and F-Score of the combined model (both classifiers) on the complete data set (excluding the extra literal examples). Note that the results for the set-ups involving iterative training set enlargement are optimistic: since we do not have a separate development set, we report the optimal performance achieved during the first seven iterations. In a real set-up, when the optimal number of iterations is chosen on the basis of a separate data set, the results may be lower. The table also shows the majority class baseline (*Base*$_{maj}$), and the overall performance of the unsupervised model (*unsup*) and the supervised model when trained in 10-fold cross-validation mode (*super 10CV*).

Model	Prec$_l$	Rec$_l$	F-Score$_l$	Acc.
Base$_{maj}$	-	-	-	78.25
unsup.	50.04	69.72	58.26	78.38
combined	83.86	45.82	59.26	86.30
combined+boost	70.26	62.76	66.30	86.13
combined+it*	85.68	46.52	60.30	86.68
combined+boost+it*	71.86	66.36	69.00	87.03
super. 10CV	85.80	66.60	75.00	90.34

Table 3: Results for different classifiers; $*$ indicates best performance (optimistic)

It can be seen that the combined classifier is 8% more accurate than both the majority baseline and the unsupervised classifier. This amounts to an error reduction of over 35% (the difference is statistically significant, χ^2 test, $p << 0.01$). While the F-Score of the unboosted combined classifier is comparable to that of the unsupervised one, boosting the literal class leads to a 7% increase, due to a significantly increased recall, with no significant drop in accuracy. These results show that complementing the unsupervised classifier with a supervised one, can lead to tangible performance gains. Note that the accuracy of the combined classifier, which uses no manually labelled training data, is only 4% below that of a fully supervised classifier; in other words, we do not lose much by starting with an automatically labelled data set. Iterative enlargement of the training set can lead to further improvements, especially when combined with boosting to reduce the classifier bias.

To get a better idea of the effect of training set enlargement, we plotted the accuracy and F-Score of the combined classifier for a given number of iterations with boosting (Figure 1) and without (Figure 2). It can be seen that enlargement has a noticeable positive effect if combined with boosting. If the literal class is not boosted, the increasing bias of the classifier seems to outweigh most of the positive effects from the enlarged training set. Figure 1 also shows that the best performance is obtained after a relatively small number of iterations (namely two), as expected.[15] With more iterations the performance decreases again. However, it decays rel-

[15] Note that this also depends on the confidence threshold. For example, if a threshold of 5% is chosen, more iterations may be required for optimal performance.

atively gracefully and even after seven iterations, when more than 40% of the data are classified by the unsupervised classifier, the combined classifier still achieves an overall performance that is significantly above that of the unsupervised classifier (84.28% accuracy compared to 78.38%, significant at $p << 0.01$). Hence, the combined classifier seems not to be very sensitive to the exact number of iterations and performs reasonably well even if the number of iterations is sub-optimal.

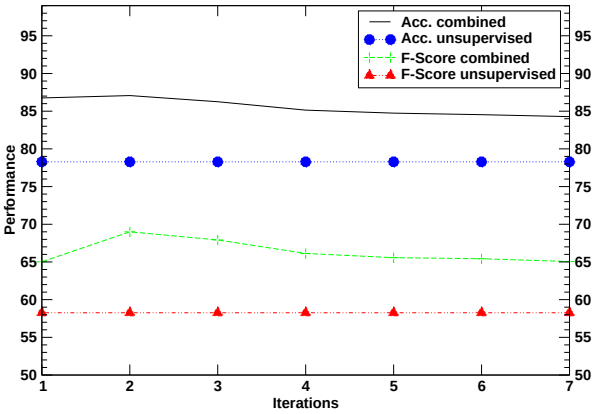

Figure 1: Accuracy and literal F-Score on complete data set after different iterations with boosting of the literal class

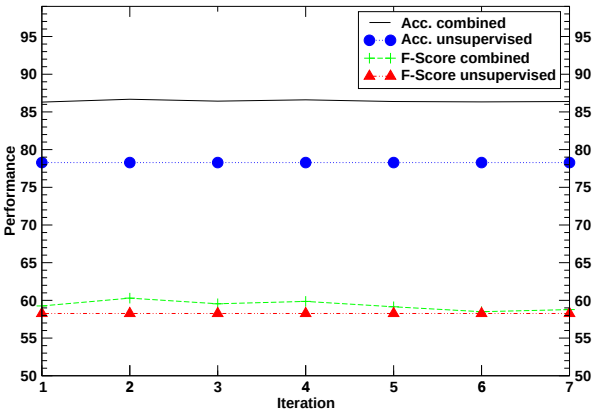

Figure 2: Accuracy and literal F-Score on complete data set after different iterations without boosting of the literal class

Figure 3 shows how the training set increases as the process goes on[16] and how the number of mis-classifications in the training set develops. Interestingly, when going from the first to the second iteration the training set nearly doubles (from 396 to 669 instances), while the proportion of errors is also reduced by a third (from 7% to 5%). Hence, the training set does not only grow but the proportion of noise in it decreases, too. This shows that our conservative enlargement strategy is fairly successful in selecting correctly labelled examples. Only at later stages, when the classifier bias takes over, does the proportion of noise increase again.

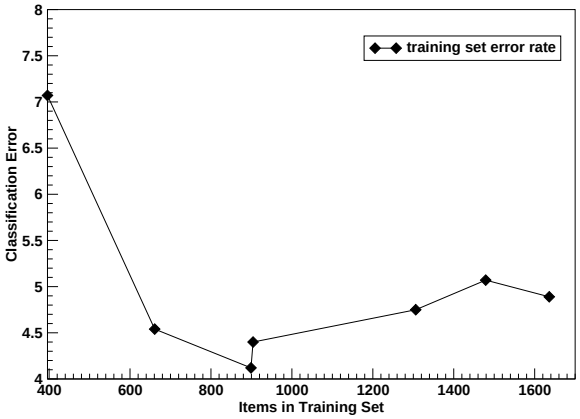

Figure 3: Training set size and error in training set at different iterations

7 Conclusion

We presented a two-stage classification approach for distinguishing literal and non-literal use of idiomatic expressions. Our approach complements an unsupervised classifier, which exploits information about the cohesive structure of the discourse, with a supervised classifier. The latter can make use of a range of features and therefore base its classification decision on additional properties of the discourse, besides lexical cohesion. We showed that such a combined classifier can lead to a significant reduction of classification errors. Its performance can be improved further by iteratively increasing the training set in a bootstrapping loop and by adding additional examples of the literal class, which is typically the minority class. We found that such examples can be obtained automatically by extracting non-canonical variants of the target idioms from an unlabelled corpus.

Future work should look at improving the supervised classifier, which so far has an accuracy of 90%. While this is already pretty good, a more sophisticated model might lead to further improvements. For example, one could experiment with linguistically more informed features. While our initial studies in this direction were negative, careful feature engineering might lead to better results.

Acknowledgements

This work was funded by the Cluster of Excellence "Multimodal Computing and Interaction".

[16] Again, we disregard the extra literal examples here.

References

Timothy Baldwin, Colin Bannard, Takaaki Tanaka, and Dominic Widdows. 2003. An empirical model of multiword expression decomposability. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*.

Colin Bannard. 2007. A measure of syntactic flexibility for automatically identifying multiword expressions in corpora. In *Proceedings of the ACL-07 Workshop on A Broader Perspective on Multiword Expressions*.

Julia Birke and Anoop Sarkar. 2006. A clustering approach for the nearly unsupervised recognition of nonliteral language. In *Proceedings of EACL-06*.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of COLT-98*.

Rudi L. Cilibrasi and Paul M.B. Vitanyi. 2007. The Google similarity distance. *IEEE Trans. Knowledge and Data Engineering*, 19(3):370–383.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurements*, 20:37–46.

Paul Cook, Afsaneh Fazly, and Suzanne Stevenson. 2007. Pulling their weight: Exploiting syntactic forms for the automatic identification of idiomatic expressions in context. In *Proceedings of the ACL-07 Workshop on A Broader Perspective on Multiword Expressions*.

Afsaneh Fazly and Suzanne Stevenson. 2006. Automatically constructing a lexicon of verb phrase idiomatic combinations. In *Proceedings of EACL-06*.

Afsaneh Fazly, Paul Cook, and Suzanne Stevenson. 2009. Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics*, 35(1):61–103.

M.A.K. Halliday and R. Hasan. 1976. *Cohesion in English*. Longman House, New York.

Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal*, 6(5):429–450.

Graham Katz and Eugenie Giesbrecht. 2006. Automatic identification of non-compositional multiword expressions using latent semantic analysis. In *Proceedings of the ACL/COLING-06 Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*.

Anh-Cuong Le, Akira Shimazu, and Le-Minh Nguyen. 2006. Investigating problems of semi-supervised learning for word sense disambiguation. In *Proc. ICCPOL-06*.

Dekang Lin. 1999. Automatic identification of non-compositional phrases. In *Proceedings of ACL-99*, pages 317–324.

Vincent Ng and Claire Cardie. 2003. Weakly supervised natural language learning without redundant views. In *Proc. of HLT-NAACL-03*.

Susanne Riehemann. 2001. *A Constructional Approach to Idioms and Word Formation*. Ph.D. thesis, Stanford University.

Caroline Sporleder and Linlin Li. 2009. Unsupervised recognition of literal and non-literal use of idiomatic expressions. In *Proceedings of EACL-09*.

Aline Villavicencio, Valia Kordoni, Yi Zhang, Marco Idiart, and Carlos Ramisch. 2007. Validation and evaluation of automatically acquired multiword expressions for grammar engineering. In *Proceedings of EMNLP-07*.